

(3) 9

PATENT ABSTRACTS OF JAPAN

(11)Publication number : 09-269912

(43)Date of publication of application : 14.10.1997

(51)Int.Cl.

G06F 12/00

G06F 12/00

(21)Application number : 08-080179

(71)Applicant : CANON INC

(22)Date of filing : 02.04.1996

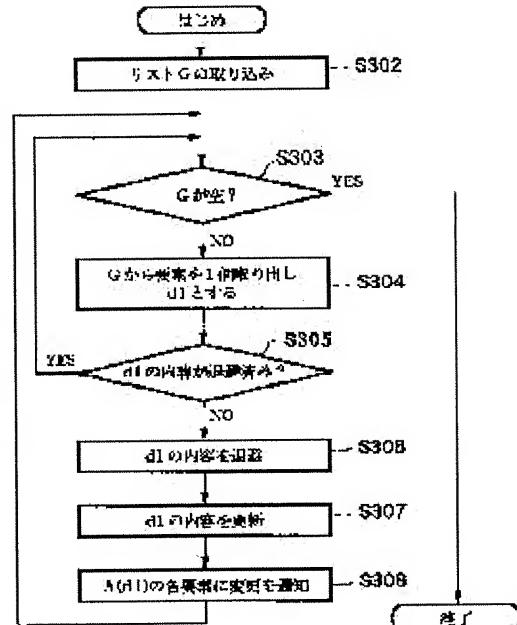
(72)Inventor : YOSHIMOTO MASAHIKO

(54) INFORMATION PROCESSING METHOD AND INFORMATION PROCESSOR

(57)Abstract:

PROBLEM TO BE SOLVED: To provide an information processing method and an information processor, which can facilitate an interactive processing between users at the time of referring to common data between the plural application software.

SOLUTION: This information processing method is to control the reference of common data by the plural applications. Correction/addition data against common data based on the request of a first application is stored. Common data is updated (S305) based on correction/addition data stored based on the committing request of the first application. Then, information on updated common data is informed to the application referring to other common data (S306).



LEGAL STATUS

[Date of request for examination]

[Date of sending the examiner's decision of rejection]

[Kind of final disposal of application other than the examiner's decision of rejection or application converted registration]

[Date of final disposal for application]

[Patent number]

[Date of registration]

[Number of appeal against examiner's decision of rejection]

[Date of requesting appeal against examiner's decision of rejection]

[Date of extinction of right]

(2)

(19)日本国特許庁 (JP)

(12) 公開特許公報 (A)

(11)特許出願公開番号

特開平9-269912

(43)公開日 平成9年(1997)10月14日

(51)Int.Cl. ⁶ G 0 6 F 12/00	識別記号 5 3 3 5 1 8	序内整理番号 F I G 0 6 F 12/00	技術表示箇所 5 3 3 F 5 1 8 A
---	------------------------	--------------------------------	------------------------------

審査請求 未請求 請求項の数8 OL (全8頁)

(21)出願番号 特願平8-80179	(71)出願人 000001007 キヤノン株式会社 東京都大田区下丸子3丁目30番2号
(22)出願日 平成8年(1996)4月2日	(72)発明者 吉本 雅彦 東京都大田区下丸子3丁目30番2号 キヤ ノン株式会社内

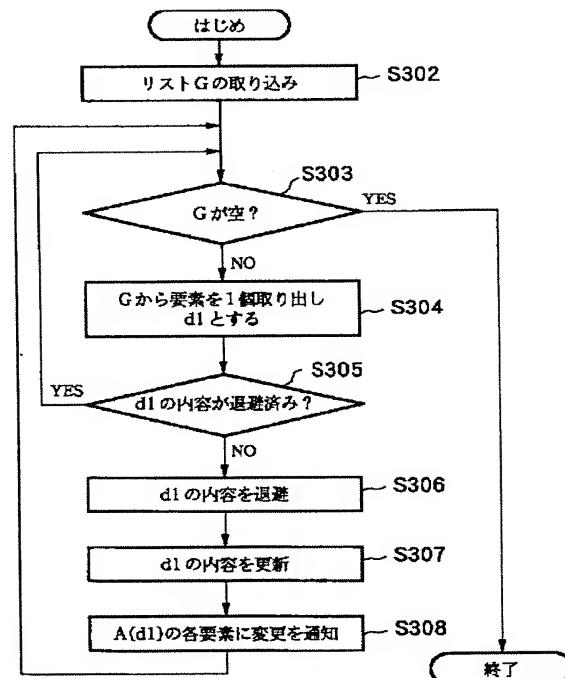
(74)代理人 弁理士 大塚 康徳 (外1名)

(54)【発明の名称】 情報処理方法とその装置

(57)【要約】

【課題】 複数のアプリケーションソフトウェア間で共有データを参照する場合、ユーザ間の対話的処理を容易に実現できる情報処理方法とその装置を提供する。

【解決手段】 複数のアプリケーションが共有データを参照する制御を行う情報処理方法であって、第1のアプリケーションの要求に基づく共有データに対する修正／追加データを格納する。そして、第1のアプリケーションのコミット要求に基づいて、格納された修正／追加データに基づいて、前記共有データを更新する(S305)。そして、更新された共有データに関する情報を他の前記共有データを参照中のアプリケーションに通知する(S306)。



【特許請求の範囲】

【請求項1】 複数のアプリケーションが共有データを参照する制御を行う情報処理方法であって、
第1のアプリケーションの要求に基づく共有データに対する修正／追加データを格納する格納工程と、
前記第1のアプリケーションのコミット要求に基づいて、前記格納工程で格納された修正／追加データに基づいて、前記共有データを更新する更新工程と、
前記更新工程で更新された共有データに関する情報を他の前記共有データを参照中のアプリケーションに通知する通知工程とを備えることを特徴とする情報処理方法。

【請求項2】 前記第1のアプリケーションのアボード要求に基づいて、前記更新工程で更新された共有データを、前記格納工程で格納される以前のデータに復帰させる復帰工程をさらに備えることを特徴とする請求項1に記載の情報処理方法。

【請求項3】 前記共有データをアクセスするアプリケーションを検知する検知工程と、
前記検知工程で検知されたアプリケーションがアクセスするデータに対して、当該アプリケーションが共有ロックまたは排他ロックを確保していない場合、当該アプリケーションが前記更新工程を行うに先だって、共有ロックを設定する設定工程とをさらに備えることを特徴とする請求項1に記載の情報処理方法。

【請求項4】 前記復帰工程はさらに、前記更新工程で更新された共有データを、前記格納工程で格納される以前のデータに復帰させる旨の情報を、前記共有データを使用中の他のアプリケーションに通知する復帰通知工程を備えることを特徴とする請求項2に記載の情報処理方法。

【請求項5】 複数のアプリケーションが共有データを参照する制御を行う情報処理装置であって、
第1のアプリケーションの要求に基づく共有データに対する修正／追加データを格納する格納手段と、
前記第1のアプリケーションのコミット要求に基づいて、前記格納手段で格納された修正／追加データに基づいて、前記共有データを更新する更新手段と、
前記更新手段で更新された共有データに関する情報を他の前記共有データを参照中のアプリケーションに通知する通知手段とを備えることを特徴とする情報処理装置。

【請求項6】 前記第1のアプリケーションのアボード要求に基づいて、前記更新手段で更新された共有データを、前記格納手段で格納される以前のデータに復帰させる復帰手段をさらに備えることを特徴とする請求項5に記載の情報処理装置。

【請求項7】 前記共有データをアクセスするアプリケーションを検知する検知手段と、
前記検知手段で検知されたアプリケーションがアクセスするデータに対して、当該アプリケーションが共有ロックまたは排他ロックを確保していない場合、当該アプリ

ケーションが前記更新工程を行うに先だって、共有ロックを設定する設定手段とをさらに備えることを特徴とする請求項5に記載の情報処理装置。

【請求項8】 前記復帰手段はさらに、前記更新手段で更新された共有データを、前記格納手段で格納される以前のデータに復帰させる旨の情報を、前記共有データを使用中の他のアプリケーションに通知する復帰通知手段を備えることを特徴とする請求項7に記載の情報処理方法。

【発明の詳細な説明】**【0001】**

【発明の属する技術分野】 本発明は、情報処理方法とその装置、特に、トランザクション間の並行性を向上させる情報処理方法とその装置に関する。

【0002】

【従来の技術】 従来より、データベースシステムにおけるトランザクション処理においては、ロック機構を用いた並行制御が用いられてきた。このロックの種類としては一般に、データの更新を行う際に必要となる排他ロックと、データの参照を行う際に必要となる共有ロックがある。

【0003】 さらにロックの獲得と解放は、2フェーズロックキングと呼ばれる方式、すなわち、トランザクション間の相互干渉を防止し、結果として、一貫性のあるスケジューリングを保証する方式に従って行われるのが通常であった。

【0004】

【発明が解決しようとする課題】 しかしながら、従来の方式では、対話的な操作環境を有するアプリケーションには適していないという問題点があった。例えば、あるユーザの操作により、データの更新を目的としてトランザクションが起動されたとき、該データに対しては排他ロックの獲得が必要である。しかし、そのためには、他のトランザクションによって、排他、もしくは共有ロックの確保が行なわれていないことが必要となる。

【0005】 これによれば、以下のような問題点が生じる。すなわち、上記データの内容が、例えば、ウインドウシステム上のウインドウ内に表示されている場合、ウインドウの内容の一貫性を保証するためには、共有ロックの獲得が必要であるが、これによれば、他のトランザクションによる排他ロックの獲得は不可能であり、ウインドウに表示されている間のデータの更新を行うことができなくなる。

【0006】 一方、一貫性の保証を放棄して、共有ロックの確保に統いてデータの読み出しを行い、該データの内容をアプリケーション内のメモリ領域に複写したのち、該獲得した共有ロックを解放すれば、他のトランザクションによるデータの更新は可能になる。しかし、この場合であっても、排他ロックが確保されている間は、後続するトランザクションによる共有ロックの確保は不可能

である。この結果、排他ロックが確保される以前に読み出しを終えたアプリケーションにおいては、必ずしも一貫性の保証されないデータが保持され、他方、排他ロックが確保された以後に読み出しを試みようとするアプリケーションにおいては、排他ロックが解放されるまでの間、予測できない待ち時間が生じることになる。これらの振舞いは、複数のユーザが協調しながら、データを共有する分散環境においては、著しく不適当である。本発明は、以上説明した問題点、すなわち、従来のトランザクション処理、特に並行制御機構を用いて、複数ユーザ間で共有されるデータを用いた対話的処理の実現が困難であったという問題点を解決する。

【0007】本発明は、上記従来例に鑑みてなされたもので、複数のアプリケーションソフトウェア間で共有データを参照する場合、従来のトランザクション処理機構にとっては、対話的な処理の実現が困難であったという問題点を解決した情報処理方法とその装置を提供することを目的とする。

【0008】

【課題を解決するための手段】上記目的を達成するため、本発明の情報処理方法とその装置は以下の構成を備える。即ち、複数のアプリケーションが共有データを参照する制御を行う情報処理方法であって、第1のアプリケーションの要求に基づく共有データに対する修正／追加データを格納する格納工程と、前記第1のアプリケーションのコミット要求に基づいて、前記格納工程で格納された修正／追加データに基づいて、前記共有データを更新する更新工程と、前記更新工程で更新された共有データに関する情報を他の前記共有データを参照中のアプリケーションに通知する通知工程とを備える。

【0009】また、別の発明は、複数のアプリケーションが共有データを参照する制御を行う情報処理装置であって、第1のアプリケーションの要求に基づく共有データに対する修正／追加データを格納する格納手段と、前記第1のアプリケーションのコミット要求に基づいて、前記格納手段で格納された修正／追加データに基づいて、前記共有データを更新する更新手段と、前記更新手段で更新された共有データに関する情報を他の前記共有データを参照中のアプリケーションに通知する通知手段とを備える。

【0010】

【発明の実施の形態】はじめに、本発明の実施の形態の情報処理方法とその装置のポイントを要約した後に、その詳細な説明に入るものとする。本発明の実施の形態の情報処理方法のポイントは、データベースの所望のデータ領域(D1)に対して共有ロックをかけずにアクセスしているアプリケーションソフトウェア(AP1)に対して、別のアプリケーションソフトウェア(AP2)が上述のデータ領域(D1)を更新するトランザクションの途中で、更新内容をコミットした時に、その更新が発

生したことを通知する。そして、別のアプリケーションソフトウェア(AP2)が、その後、アボードした場合、即ち、更新を無効として、データ領域(D1)の元の内容に戻す処理を行う。そして、その元の内容に戻した処理を行ったことを、再び、アプリケーションソフトウェア(AP1)に対して通知する。この様な処理構成を備えることによって、複数ユーザ間で共有されるデータを用いた対話的処理を容易に行うことができる。

【0011】尚、本処理構成では、上述のアプリケーションソフトウェア(AP1)に対しては、アプリケーションソフトウェア(AP1)が参照したいデータ領域に対する共有ロックが係っているか否かにかかわらず、データ参照操作を許可する。

【実施例1】以下、本発明の実施の形態の情報処理方法とその装置について、図を用いて説明する。なお、本発明を実施する装置としては、周知のワークステーション、もしくは、パーソナルコンピュータでよい。

【0012】図1は、所定のアプリケーションソフトウェアからのデータ参照要求を処理する手順を示すフローチャートである。ここで、このアプリケーションソフトウェアの識別子を”a”，データの識別子を”d”とする。また、図2は、図1のフローチャートが示す処理の基本要素を記述した図である。図2は、アプリケーションソフトウェアa(1000)、アプリケーションソフトウェアb(1001)、...、アプリケーションソフトウェアm(1002)が本情報処理装置内に存在している例を示している。また、各アプリケーションソフトウェアが、DB(データベース)(3000)中のデータ領域”d”を共通にアクセスする場合を示している。

【0013】アプリケーションソフトウェアa(1000)に対応する参照データ識別子リストD(a)(2000)は、アプリケーションソフトウェアa(1000)がDB(3000)のどのデータをアクセス中であるかを示すためのもので、参照中の各データの識別子を格納したリストである。データ参照アプリケーションリストA(d)(2001)は、DB(3000)中のデータ”d”をアクセス中の各アプリケーションソフトウェアの識別子を格納したリストである。

【0014】これらの参照データ識別子リストD(a)(2000)とデータ参照アプリケーションリストA(d)(2001)は、図7を参照して後述するメモリ202の一部の領域にアサインされている。以下、図2を図1のフローチャートの示す処理手順の理解を容易にするための補助図として参照しながら、アプリケーションソフトウェアからのデータ参照要求を処理する手順を説明する。この例では、排他ロックや共有ロックをかけずにデータベースをアクセスする処理に関して説明する。図1のフローチャートの示す処理は、所定のトランザクションマネージャ(TM)などによって、実行される。

【0015】まず、ステップS102では、アプリケ

ションソフトウェア a から、TM に対して、データベースのデータ d を参照する要求があると、TM は、アプリケーションソフトウェア a が参照しているデータ識別子のリスト D (a) を取り出す。ステップ S 103 では、D (a) に d が含まれているかどうか判定する。そして、含まれていれば、アプリケーションソフトウェア a は、既にデータ d の参照処理を起動したものとみなして、直ちに処理を終了する。逆に、含まれていなければ、次のステップ S 104 に移行する。

【0016】ステップ S 104 では、d を参照しているアプリケーションソフトウェアのリスト A (d) を取り出す。ステップ S 105 では、A (d) には a は含まれていないので、A (d) に a を追加する。ステップ S 106 では、識別子 d の指すデータ内容をアプリケーションソフトウェア a に転送する。

【0017】ステップ S 107 では、D (a) に d を追加して、処理を終了する。以上、アプリケーションソフトウェア a からデータベースに対するデータ参照要求を処理する手順を説明した。尚、図 1において、2つのリスト D (a) と A (d) との一貫性は、公知の手法によって常に保たれているものとする。

【0018】また、ステップ S 102～S 104、および、ステップ S 105～S 107 の処理順序は、実施しているデータ共有機構の実装形態に依存するものであり、本発明は、この処理順序に制限されるものではない。図 1 に示した手順の逆操作、すなわち、データ参照の解除については、図 1 と全く同様である。すなわち、アプリケーションソフトウェア a とデータ d について、D (a) に d が含まれると、D (a) から d を削除するとともに、A (d) から a を削除すればよい。

【0019】図 3 は、TM が、図 1 の手順に従って共有データに対する参照処理を開始した後に、各アプリケーションからアクセスされるデータベースのデータに対する共有ロックの設定処理の説明を行う。図 4 は、共有ロックの設定対象を管理するリスト L, L' の構成例を示す図である。

【0020】図 4 の共有ロック要求識別子リスト L (4 000) は、TM が生成したもので、各アプリケーションソフトウェアからの共有ロックの獲得処理を要求するデータ識別子のリストである。これは、TM が、データベースへの各アプリケーションからのアクセスを監視して生成した、データベースへアクセスのあったデータのリスト、即ち、上述した

D (x) ; x = a, b, . . . c

に等価のリストであるか、または、D (x) のうち、アプリケーションソフトウェアから共有ロック要求が行われたデータのリストである。

【0021】ここで、この共有ロック要求識別子リスト L (4 000) は、図 7 を参照して後述するメモリ 202 の一部の領域にアサインされている。図 3 のステップ

S 202 では、上述の共有ロック要求識別子リスト L (4 000) の要素であって、既に、共有ロック、または、排他ロックが確保されているものを取り除いたリスト L' (非ロックデータ識別子リスト) を生成する。

【0022】ステップ S 203 では、リスト L' が空かどうかをチェックする。そして、空でなければ、ステップ S 204 へ進む。空であれば、処理を終了する。ステップ S 204 では、リスト L' の各要素に対する共有ロックの確保を行い、処理を終了する。以上の処理を TM が繰り返すことによって、共有ロックの必要なデータに対して、適宜、共有ロックをかけていくことができる。

【0023】TM は、また上述の処理と共に、データベースへの各アプリケーションからのアクセスの内のデータ更新の発生を監視して、データ更新の発生したデータの識別子を羅列したデータリスト G を生成する。次に、図 5 は、各アプリケーションがデータ更新 (コミット) を行うトランザクションにおける TM の処理手順を示すフローチャートである。

【0024】まず、ステップ S 302 では、リスト G のデータを取り込む。ステップ S 303 では、リスト G が空かどうかチェックして、空であれば処理を終了する。他方、空でなければステップ S 304 へ進む。ステップ S 304 では、リスト G から要素を 1 つ取り出して d 1 とする。ステップ S 305 では、d 1 の内容が既に退避済みかどうかチェックする。そして、退避済みであれば、ステップ S 303 に戻る。逆に、退避済みでなければ、ステップ S 306 に進む。

【0025】ステップ S 306 では、d 1 の内容を一時的に退避する。ステップ S 307 では、d 1 の内容を更新する。ステップ S 308 では、d 1 の参照を行っているリスト A (d 1) の各アプリケーションソフトウェアに対し変更の通知を行う。そして、ステップ S 303 に戻り、同様の処理を繰り返す。

【0026】以上の処理によって、所定のアプリケーションがデータ更新した情報を、参照中の他のアプリケーションに知らせることができる。尚、ステップ S 306 における d 1 の内容の退避領域は、最終的にトランザクションが終了した時点で解放される。ここで、トランザクションが最終的に終了するとは、コミットまたはアボートのいずれかである。ここで、コミットとは、更新した内容を確定する処理である。また、アボートとは、更新した内容を確定せずに廃棄する、言い換えれば、更新前の状態に戻す処理を意味する。

【0027】従って、コミットであった場合には、單に、退避領域が解放される。一方、アボートであった場合には、d 1 の退避領域が空でなければ、一時的に行われた更新処理の内容が破棄され、d 1 の内容として退避領域の内容が再設定される。この時、A (d 1) が空でなければ、ステップ S 308 と同様に A (d 1) の各要素に対して変更の通知が行われる。

【0028】以上、説明した処理手順を用いれば、例えば、あるアプリケーションソフトウェアa1とa2において、効果的なデータ共有を行うことができる。これを図6を参照して説明する。図6に示すように、データベースにデータd t 1, d t 2, d t 3, d t 4があるとし、アプリケーションソフトウェアa1とa2がそれぞれ図1の手順に従って、d t 1, d t 2, d t 3、および、d t 1, d t 2, d t 4の参照手順を開始したとする。

【0029】ここで、アプリケーションソフトウェアa1とa2が、それぞれd t 1とd t 2の更新を行なうためにトランザクションを開始したとする。このとき、図3に示した処理手順において、共有ロックを確保するデータ識別子のリストは、アプリケーションソフトウェアa1についてはd t 3であり、a2についてはd t 4を含むものとする。

【0030】図3に示した手順によって上記の共有ロックが確保されれば、何らかの時点でa1とa2はそれぞれd t 1とd t 2に関する排他ロックを確保することになるが、a1とa2間のロックの競合が存在しないため、他の外乱要因がなければこれらはともに成功する。この段階で、図5で示した手順に従って一時的な内容更新を行なえば、アプリケーションソフトウェアa1とa2は、それぞれ、他方が行った更新が直ちに通知され、それに従って処理を行うことが可能である。

【0031】なお、一般には、データ更新トランザクションの途中で、他のトランザクションによる一時的な更新通知が行われるのは好ましくない。これは、前者のトランザクションにおける一時的な更新内容の評価によっては、最終的に、コミットを行う段階で一貫性が保証されない可能性があるためである。従って、通常のトランザクション開始時には、図1の手順で参照を開始した全ての共有データについて図3に示した手順を適用するのが適当である。

【0032】さらに、トランザクションとしては、データの更新を行わない類のトランザクションもあるが、これについても図3に示した手順を適用することは可能である。この場合のトランザクションの目的は、トランザクションの間に他のトランザクションに更新を禁止することである。本実施の形態においては、データ参照処理における共有ロックの確保は不用であって、むしろ不必要なロックによりトランザクション間の競合を回避するために、読み出しトランザクションは必要最小限にとどめるのが適当である。

【実施の形態2】以上説明した実施の形態においては、トランザクションにおけるロックの獲得・解放処理を2フェーズで行なるものとしていた。

【0033】しかしながら、本実施の形態に係る更新処理と参照処理が共存し、なおかつ、一時的な更新処理を支援するようなシステムにおいては、いわゆる「seriali

zability」を犠牲にしても、データの共有を許すのが適當である。これを実現するために、明示的なロック解放命令をアプリケーションに提供する。図7は、明示的なロック解放命令を受信した時のTMの処理手順を示すフローチャートである。

【0034】尚、本フローチャートにおいて、排他ロックを解放して共有ロックを確保する手順を示すが、排他ロックの解放処理や共有ロックの解放処理についても同様である。ただし、共有ロックの解放処理については、後述する一時的な更新処理は行なわれない。ステップS401では、明示的にロックの解放を行うデータ識別子d2を受信する。

【0035】ステップS402では、d2に関する排他ロックが既に確保されているかどうかをチェックする。そして、排他ロックの確保が行われていなければ処理を終了する。さもなくば、ステップS403へ進む。ステップS403では、d2の内容を変更したかどうかを判定する。そして、変更を行なっている場合には、ステップS404へ進む。変更がなければ、ステップS405に進む。

【0036】ステップS404では、図5に示した手順に従って一時的に更新内容を反映させる。ステップS405では、排他ロックを共有ロックに変更して、処理を終了する。尚、ステップS405における処理は不可分に行なわれるものとし、かつ、該ロックの解放を待っている他のトランザクションに優先して行なわれるものとする。

【0037】図7においては、2フェーズロックに従わない個々のトランザクションについて処理手順を示した。しかし一般には、入れ子になったトランザクション処理が行われることがあり、上位のトランザクションが2フェーズロックに従うにもかかわらず、下位のサブトランザクションにおいて図7の手順を行うと矛盾が生じる。

【0038】これを防ぐために、上位トランザクションのいずれかがロックの獲得・解放処理を2フェーズロックに従う場合には、サブトランザクションにおけるロックの獲得・解放処理は指定かかわらず2フェーズで行なうものとする。この場合、上記明示的なロック解放命令は遅延評価され、最終的には2フェーズロックに従わない上位トランザクションに戻った時点で評価される。

【0039】次に、図8は、上述した各フローチャートの処理を実行する情報処理装置の構成例を示す図である。この図で、CPU200は、本情報処理装置全体の制御を、メモリ202に格納された各種プログラムを読み出し、解釈し、実行することで行なう。メモリ202には、上述した各フローチャートに対応するプログラムが格納され、また、データベースが格納されている。

【0040】キーボード203とポインティングデバイス204は、コマンドやデータの入力を行う。ディスプ

レイモニタ201は、CPU200での処理結果や、キーボード203とポインティングデバイス204から入力したコマンドやデータを表示する。なお、本発明は、複数の機器から構成されるシステムに適用しても、一つの機器からなる装置に適用してもよい。

【0041】また、本発明の目的は、前述した実施形態の機能を実現するソフトウェアのプログラムコードを記録した記憶媒体を、システムあるいは装置に供給し、そのシステムあるいは装置のコンピュータ（またはCPUやMPU）が記憶媒体に格納されたプログラムコードを読み出し実行することによっても、達成されることは言うまでもない。

【0042】この場合、記憶媒体から読み出されたプログラムコード自体が前述した実施形態の機能を実現することになり、そのプログラムコードを記憶した記憶媒体は本発明を構成することになる。プログラムコードを供給するための記憶媒体としては、例えば、フロッピディスク、ハードディスク、光ディスク、光磁気ディスク、CD-ROM、CD-R、磁気テープ、不揮発性のメモリカード、ROMなどを用いることができる。

【0043】また、コンピュータが読み出したプログラムコードを実行することにより、前述した実施形態の機能が実現されるだけでなく、そのプログラムコードの指示に基づき、コンピュータ上で稼働しているOS（オペレーティングシステム）などが実際の処理の一部または全部を行い、その処理によって前述した実施形態の機能が実現される場合も含まれることは言うまでもない。

【0044】さらに、記憶媒体から読み出されたプログラムコードが、コンピュータに挿入された機能拡張ボードやコンピュータに接続された機能拡張ユニットに備わるメモリに書き込まれた後、そのプログラムコードの指示に基づき、その機能拡張ボードや機能拡張ユニットに備わるCPUなどが実際の処理の一部または全部を行い、その処理によって前述した実施形態の機能が実現される場合も含まれることは言うまでもない。

【0045】本発明を上記記憶媒体に適用する場合、その記憶媒体には、先に説明したフローチャートに対応するプログラムコードを格納することになるが、簡単に説明すると、上述の各フローチャートに対応する各プログラムモジュールを記憶媒体に格納することになる。以上説明したように、本実施の形態の情報処理方法では、アプリケーションソフトウェアに対して、共有ロックの獲得の有無にかかわらずデータ参照操作を許可する。また、該データ参照操作を行っているアプリケーションソフトウェアがデータ更新処理を目的とするトランザクシ

ョンを開始する際に、そのトランザクションの起動に先だって、上記データ参照操作が起動されている一連の共有データに対して共有ロックを確保するかどうかを指定する。そして、該データ更新処理を目的とするトランザクションの処理の途中で一時的に更新内容をコミットした場合、そのコミット処理でデータを変更したことを知らせる変更通知を、上記データ参照操作を起動している一連のアプリケーションソフトウェアに対して通知する。

10 【0046】また、上記トランザクションが最終的にアボートされた場合には、上記コミット処理に対応する逆操作を行って、元のデータを回復する。そして、その逆操作における変更通知を、上記データ参照操作を起動している一連のアプリケーションソフトウェアに対して通知する。尚、上記データ更新処理を目的とするトランザクションは、ロックの獲得・解放処理を2フェーズロックに従って行うかどうかを指定することができる。

【0047】以上の基本処理構成によって、複数ユーザー間で共有されるデータを用いた対話的処理を容易に実現することができる。

【0048】

【発明の効果】以上説明したように本発明によれば、複数のアプリケーションソフトウェア間で共通データを参照する場合、従来のトランザクション処理機構にとっては、対話的な処理の実現が困難であったという問題点を解決できる。

【図面の簡単な説明】

【図1】本発明に係る一実施の形態の各アプリケーションからデータベースの所定のデータに対するデータ参照要求を処理する手順を示すフローチャートである。

【図2】図1の処理を説明するための図である。

【図3】共有ロックの獲得処理手順を示すフローチャートである。

【図4】共有ロック要求データ識別子リストと非ロックデータ識別子リストの構成例を示す図である。

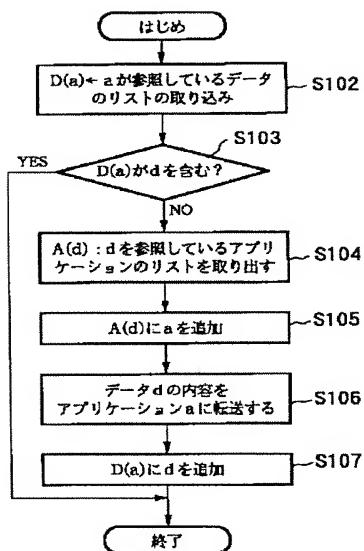
【図5】トランザクションの途中で一時的にデータ更新内容を、所定のアプリケーションに通知する処理する処理手順を示すフローチャートである。

【図6】第1の実施の形態の処理例を説明するための図である。

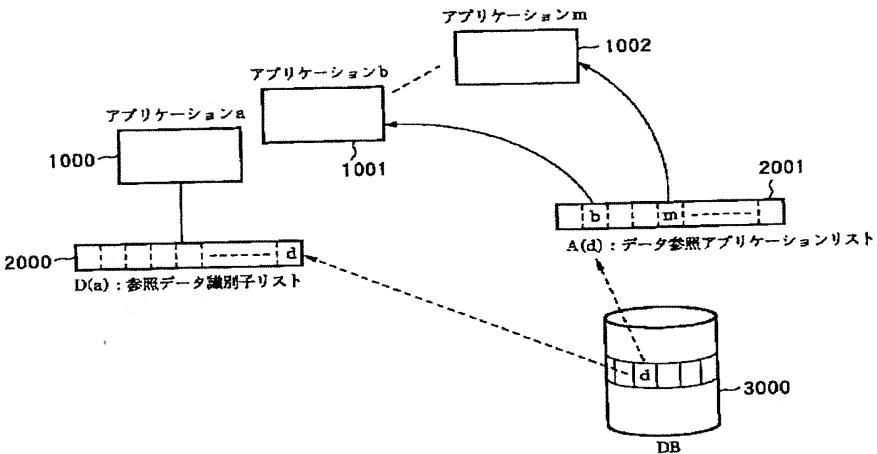
【図7】明示的にロック解放を行う処理手順を示すフローチャートである。

【図8】本実施の形態の情報処理を実行するハードウェア構成の一例を示す図である。

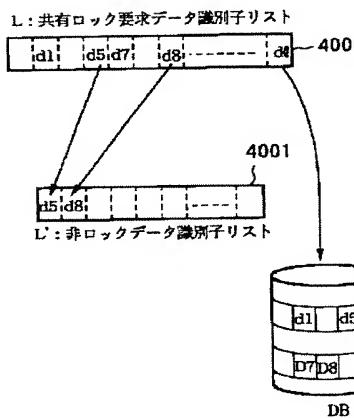
【図1】



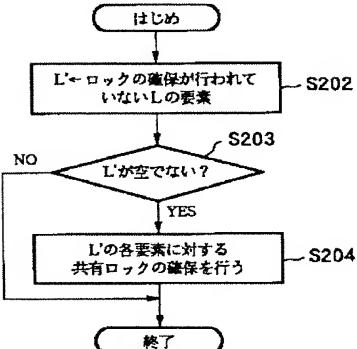
【図2】



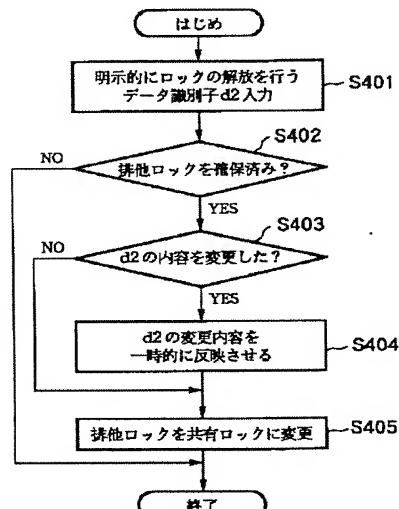
【図4】



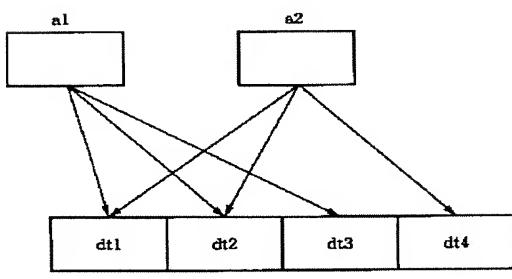
【図3】



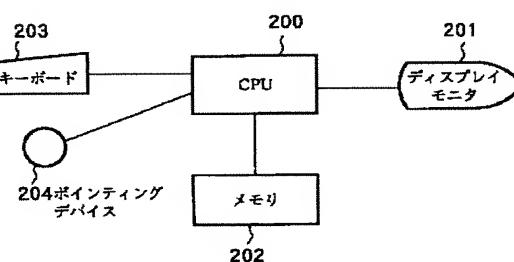
【図7】



【図6】



【図8】



【図5】

